**ASSESSMENT 10**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | 28-05-2020 | **Name:** | Sheela Golasangi |
| **Course:** | Logic Design | **USN:** | 4AL16EC068 |
| **Topic:** | Boolean equations for digital circuits, Combinational Circuits: Conversion of MUX and Decoders to logic gates, Decoder with common design of 7 segment anode display. | **Semester & Section:** | VIII  ‘B’ |
| **Github Repository:** | Sheela - Course |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of the session**  **C:\Users\india\Pictures\Screenshots\Screenshot (350).pngC:\Users\india\Pictures\Screenshots\Screenshot (353).pngC:\Users\india\Pictures\Screenshots\Screenshot (354).png** |
| **REPORT**  **Digital Arithmetic Circuits:**  The basic arithmetic circuits like Binary adder and Binary subtractor. These circuits can be operated with binary values 0 and 1.   * **Binary Adder**   The most basic arithmetic operation is addition. The circuit, which performs the addition of two binary numbers, is known as Binary adder. First, let us implement an adder, which performs the addition of two bits.   * **Half Adder**   Half adder is a combinational circuit, which performs the addition of two binary numbers A and B are of single bit. It produces two outputs sum, S & carry, C.  The Truth table of Half adder is shown below.   |  |  |  |  | | --- | --- | --- | --- | | **Inputs** | | **Outputs** | | | A | B | C | S | | 0 | 0 | 0 | 0 | | 0 | 1 | 0 | 1 | | 1 | 0 | 0 | 1 | | 1 | 1 | 1 | 0 |     From Truth table, we can directly write the Boolean functions for each output as  S=A⊕B  C=AB  We can implement the above functions with 2-input Ex-OR gate & 2-input AND gate. The circuit diagram of Half adder is shown in the following figure.  Half Adder  In the above circuit, a two input Ex-OR gate & two input AND gate produces sum, S & carry, C respectively. Therefore, Half-adder performs the addition of two bits. Full Adder Full adder is a combinational circuit, which performs the addition of three bits A, B and Cin. Where, A & B are the two parallel significant bits and Cin is the carry bit, which is generated from previous stage. This Full adder also produces two outputs sum, S & carry, Cout, which are similar to Half adder.  The Truth table of Full adder is shown below.   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Inputs** | | | **Outputs** | | | **A** | **B** | **Cin** | **Cout** | **S** | | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 1 | 0 | 1 | | 0 | 1 | 0 | 0 | 1 | | 0 | 1 | 1 | 1 | 0 | | 1 | 0 | 0 | 0 | 1 | | 1 | 0 | 1 | 1 | 0 | | 1 | 1 | 0 | 1 | 0 | | 1 | 1 | 1 | 1 | 1 |   We will get the following Boolean functions for each output after simplification.  S=A⊕B⊕Cin  Cout = AB+(A⊕B)cin  The circuit diagram of Full adder is shown in the following figure.  Full Adder   * **4-bit Binary Adder**   The 4-bit binary adder performs the addition of two 4-bit numbers. Let the 4-bit binary numbers, A=A3A2A1A0 and B=B3B2B1B0.  The block diagram of 4-bit binary adder is shown in the following figure.  Four Bit Binary Adder   * **4-bit Binary Subtractor**   The 4-bit binary subtractor produces the subtraction of two 4-bit numbers. Let the 4bit binary numbers, A=A3A2A1A0 and B=B3B2B1B0. Internally, the operation of 4-bit Binary subtractor is similar to that of 4-bit Binary adder. If the normal bits of binary number A, complemented bits of binary number B and initial carry borrow, Cin as one are applied to 4-bit Binary adder, and then it becomes 4-bit Binary subtractor. The block diagram of 4-bit binary subtractor is shown in the following figure.  4 Bit Binary Subtractor   * **Decoder**   It is a combinational circuit that has ‘n’ input lines and maximum of 2n output lines. One of these outputs will be active High based on the combination of inputs present, when the decoder is enabled. That means decoder detects a particular code. The outputs of the decoder are nothing but the min terms of ‘n’ input variables lines, when it is enabled.  **2 to 4 Decoder**  Let 2 to 4 Decoder has two inputs A1 & A0 and four outputs Y3, Y2, Y1 & Y0. The block diagram of 2 to 4 decoder is shown in the following figure.  2 to 4 Decoder  One of these four outputs will be ‘1’ for each combination of inputs when enable, E is ‘1’. The Truth table of 2 to 4 decoder is shown below.   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | **Enable** | **Inputs** | | **Outputs** | | | | | **E** | **A1** | **A0** | **Y3** | **Y2** | **Y1** | **Y0** | | 0 | x | x | 0 | 0 | 0 | 0 | | 1 | 0 | 0 | 0 | 0 | 0 | 1 | | 1 | 0 | 1 | 0 | 0 | 1 | 0 | | 1 | 1 | 0 | 0 | 1 | 0 | 0 | | 1 | 1 | 1 | 1 | 0 | 0 | 0 |   From Truth table, we can write the Boolean functions for each output as  Y3=E.A1.A0Y3=E.A1.A0  Y2=E.A1.A0′Y2=E.A1.A0′  Y1=E.A1′.A0Y1=E.A1′.A0  Y0=E.A1′.A0′Y0=E.A1′.A0′  Each output is having one product term. So, there are four product terms in total. We can implement these four product terms by using four AND gates having three inputs each & two inverters. The circuit diagram of 2 to 4 decoder is shown in the following figure.  2 to 4 Decoder Circuit Diagram  **4 to 16 Decoder:**  The block diagram of 4 to 16 decoder using 3 to 8 decoders is shown in the following figure.  4 to 16 Decoder   * **Encoder:**   It is a combinational circuit that performs the reverse operation of Decoder. It has maximum of 2n input lines and ‘n’ output lines. It will produce a binary code equivalent to the input, which is active High. Therefore, the encoder encodes 2n input lines with ‘n’ bits. It is optional to represent the enable signal in encoders.  **4 to 2 Encoder**  Let 4 to 2 Encoder has four inputs Y3, Y2, Y1 & Y0 and two outputs A1 & A0. The block diagram of 4 to 2 Encoder is shown in the following figure.  4 to 2 Encoder  At any time, only one of these 4 inputs can be ‘1’ in order to get the respective binary code at the output. The Truth table of 4 to 2 encoder is shown below.   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Inputs** | | | | **Outputs** | | | **Y3** | **Y2** | **Y1** | **Y0** | **A1** | **A0** | | 0 | 0 | 0 | 1 | 0 | 0 | | 0 | 0 | 1 | 0 | 0 | 1 | | 0 | 1 | 0 | 0 | 1 | 0 | | 1 | 0 | 0 | 0 | 1 | 1 |   From Truth table, we can write the Boolean functions for each output as  A1=Y3+Y2A1=Y3+Y2  A0=Y3+Y1A0=Y3+Y1  We can implement the above two Boolean functions by using two input OR gates. The circuit diagram of 4 to 2 encoder is shown in the following figure.  4 to 2 Encoder Circuit Diagram  The above circuit diagram contains two OR gates. These OR gates encode the four inputs with two bits.   * **Multiplexer:**   **Multiplexer** is a combinational circuit that has maximum of 2n data inputs, ‘n’ selection lines and single output line. One of these data inputs will be connected to the output based on the values of selection lines.  Since there are ‘n’ selection lines, there will be 2n possible combinations of zeros and ones. So, each combination will select only one data input. Multiplexer is also called as **Mux**. Draw AND gate using 2x1 MULTIPLEXER [AND_by_mux](http://lh6.ggpht.com/-mgC0yx_dyfo/T63m2UGL8FI/AAAAAAAAAa4/kVPNp1Iqy54/s1600-h/AND_by_mux%25255B6%25255D.jpg)  Look at the truth table of AND gate. When any of the one input is zero output is always zero (or same as that input); when the other input is one, output is dependent on the other input and is same as the other input. Using this property we can draw AND gate in four different ways using 2:1 MUX as shown in the above figure.   * **Programmable Logic Devices:**   Programmable Logic Devices PLDs are the integrated circuits. They contain an array of AND gates & another array of OR gates. There are three kinds of PLDs based on the type of arrays, which has programmable feature.   * Programmable Read Only Memory * Programmable Array Logic * Programmable Logic Array   The process of entering the information into these devices is known as programming. Basically, users can program these devices or ICs electrically in order to implement the Boolean functions based on the requirement. Here, the term programming refers to hardware programming but not software programming.   * **Basics of Threshold gate**   Let the inputs of threshold gate are X1, X2, X3,…, Xn. The corresponding weights of these inputs are W1, W2, W3,…, Wn. The symbol of Threshold gate is shown in the following figure.  Basics Threshold Gate  Threshold gate is represented with a circle and it is having ‘n’ inputs, X1 to Xn and single output, Y. This circle is made into two parts. One part represents the weights corresponding to the inputs and other part represents Threshold value, T.  The sum of products of inputs with corresponding weights is known as weighted sum. If this weighted sum is greater than or equal to Threshold value, T then only the output, Y will be equal to one. Otherwise, the output, Y will be equal to zero.  Mathematically, we can write this relationship between inputs and output of Threshold gate as below.  Y=1, if W1X1+W2X2+W3X3+...WnXn≥T Y=1,ifW1X1+W2X2+W3X3+...WnXn≥T  𝑌 = 0, otherwise.  Therefore, we can implement various logic gates and Boolean functions just by changing the values of weights and / or Threshold value, T.  There are two types of memory elements based on the type of triggering that is suitable to operate it.   * Latches * Flip-flops   Latches operate with enable signal, which is level sensitive. Whereas, flip-flops are edge sensitive. We will discuss about flip-flops in next chapter. Now, two types of latches are SR Latch & D Latch one by one.  SR Latch  SR Latch is also called as Set Reset Latch. This latch affects the outputs as long as the enable, E is maintained at ‘1’. The circuit diagram of SR Latch is shown in the following figure.  SR Latch  **Flip-flops:**  In second method, we can directly implement the flip-flop, which is edge sensitive. In this chapter, let us discuss the following flip-flops using second method.   * SR Flip-Flop * D Flip-Flop * JK Flip-Flop * T Flip-Flop   SR Flip-Flop  SR flip-flop operates with only positive clock transitions or negative clock transitions. Whereas, SR latch operates with enable signal. The circuit diagram of SR flip-flop is shown in the following figure.  SR Flip-Flop  **Shift Register**  We know that one flip-flop can store one-bit of information. In order to store multiple bits of information, we require multiple flip-flops. The group of flip-flops, which are used to hold store the binary data, is known as register.  If the register is capable of shifting bits either towards right hand side or towards left hand side is known as shift register. An ‘N’ bit shift register contains ‘N’ flip-flops. Following are the four types of shift registers based on applying inputs and accessing of outputs.   * Serial In − Serial Out shift register * Serial In − Parallel Out shift register * Parallel In − Serial Out shift register * Parallel In − Parallel Out shift register   Serial In − Serial Out SISOSISO Shift Register  The shift register, which allows serial input and produces serial output, is known as Serial In – Serial Out SISOSISO shift register. The block diagram of 3-bit SISO shift register is shown in the following figure.  SISO  This block diagram consists of three D flip-flops, which are cascaded. That means, output of one D flip-flop is connected as the input of next D flip-flop. All these flip-flops are synchronous with each other since, the same clock signal is applied to each one.   * **Finite State Machine**   We know that synchronous sequential circuit’s change affect their states for every positive or negative transition of the clock signal based on the input. So, this behavior of synchronous sequential circuits can be represented in the graphical form and it is known as state diagram.  A synchronous sequential circuit is also called as Finite State Machine FSM, if it has finite number of states. There are two types of FSMs.   * Mealy State Machine * Moore State Machine   Now, let us discuss about these two state machines one by one.  Mealy State Machine  A Finite State Machine is said to be Mealy state machine, if outputs depend on both present inputs & present states. The block diagram of Mealy state machine is shown in the following figure.  Mealy State Machine  As shown in figure, there are two parts present in Mealy state machine. Those are combinational logic and memory. Memory is useful to provide some or part of previous outputs present states as inputs of combinational logic.  So, based on the present inputs and present states, the Mealy state machine produces outputs. Therefore, the outputs will be valid only at positive or negative transition of the clock signal.  The state diagram of Mealy state machine is shown in the following figure.  State Diagram Algorithmic State Machines Basic Components of ASM charts  Following are the three basic components of ASM charts.   * State box * Decision box * Conditional output box * **Number Systems:**   If base or radix of a number system is ‘r’, then the numbers present in that number system are ranging from zero to r-1. The total numbers present in that number system is ‘r’. So, we will get various number systems, by choosing the values of radix as greater than or equal to two.  In this chapter, let us discuss about the popular number systems and how to represent a number in the respective number system. The following number systems are the most commonly used.   * Decimal Number system * Binary Number system * Octal Number system * Hexadecimal Number system   **Boolean Algebra and Logic Gates:**  In Boolean algebra the binary values are ‘0’ or ‘1’.   * Cost of the circuit. * Simple realization of a circuit.   In 1854, George Boole developed an algebraic system now called Boolean Algebra.  Boolean algebra is a system of mathematical logic. Or “It is defined with set of elements, a set of operators, and a number of axioms or postulates”.   * **Axioms and laws of Boolean Algebra**   Axioms or postulates of Boolean algebra are a set of logical expression open which we can build a set of useful theorems.  “AND operation” “OR operation” “NOT operation”  0·0 = 0 0+0 = 0 0’=1  0·1 = 0 0+1 = 1 1’=1  1·0 = 0 1+0 = 1  1·1 = 1 1+1 = 1   * **Difference between Boolean Algebra, Ordinary Algebra and Binary Number System:** * In Boolean Algebra   A+A = A A ·A = A  1+1 = 1 1 · 1 = 1   * In Ordinary Algebra   A+A = 2A A ·A = A2  1+1 = 2 1·1 = 1   * In Binary System   1+1 = 1 0 1·1 = 1     * **Laws of Boolean Algebra:**  1. Commutative Law 2. Associative Law 3. Distributive Law  * **Theorem of Boolean Algebra** * **Absorption Theorem:**   1. x+xy = x   x(1+y)  x ·1  x   * 1. x+x’y = x+y   (x+x’) (x+y)  1 (x+y)  x+y   * **MUX to Logic gates**  1. NAND, NOR – Universal gates because they can create any of the logic gates. 2. Universal gates used to create any of the logic gates. 3. MUX and Decoders are called “Universal Logic”.  * **BCD to 7 segment Decoder**   In **Binary Coded Decimal (BCD)** encoding scheme each of the decimal numbers (0-9) is represented by its equivalent binary pattern (which is generally of 4-bits).  Whereas, **Seven segment** display is an electronic device which consists of seven Light Emitting Diodes (LEDs) arranged in some definite pattern (common cathode or common anode type), which is used to display Hexadecimal numerals (in this case decimal numbers, as input is BCD i.e., 0-9).  Two types of seven segment LED display:   1. **Common Cathode Type:** In this type of display all cathodes of the seven LEDs are connected together to the ground or -Vcc(hence,common cathode) and LED displays digits when some ‘HIGH’ signal is supplied to the individual anodes. 2. **Common Anode Type:** In this type of display all the anodes of the seven LEDs are connected to battery or +Vcc and LED displays digits when some ‘LOW’ signal is supplied to the individual cathodes.   But, seven segment display does not work by directly supplying voltage to different segments of LEDs. First, our decimal number is changed to its BCD equivalent signal then BCD to seven segment decoder converts that signals to the form which is fed to seven segment display.  This BCD to seven segment decoder has four input lines (A, B, C and D) and 7 output lines (a, b, c, d, e, f and g), this output is given to seven segment LED display which displays the decimal number depending upon inputs.  https://media.geeksforgeeks.org/wp-content/cdn-uploads/bcd.png  Truth Table **–** For common cathode type BCD to seven segment decoder:  https://media.geeksforgeeks.org/wp-content/cdn-uploads/20191125170248/1221.png  **Example –**  https://media.geeksforgeeks.org/wp-content/uploads/bcd-1.png  **Explanation**  For combination where all the inputs (A, B, C and D) are zero (see Truth Table), our output lines are a = 1, b = 1, c = 1, d = 1, e = 1, f = 1 and g = 0. So 7 segment display shows ‘zero’ as output. |